Finding Teams in Graphs and its Application to Spatial Gene Cluster Discovery

Tizian Schulz[1,2], Jens Stoye[1], and Daniel Doerr[1](✉)

[1] Faculty of Technology and CeBiTec, Bielefeld University, Bielefeld, Germany
{tizian.schulz,jens.stoye,daniel.doerr}@uni-bielefeld.de
[2] International Research Training Group 1906 "Computational Methods for the Analysis of the Diversity and Dynamics of Genomes", Bielefeld University, Germany

Abstract. Gene clusters are sets of genes in a genome with associated functionality. Often, they exhibit close proximity to each other on the chromosome which can be beneficial for their common regulation. A popular strategy for finding gene clusters is to exploit the close proximity by identifying sets of genes that are consistently close to each other on their respective chromosomal sequences across several related species.
Yet, even more than gene proximity on linear DNA sequences, the spatial conformation of chromosomes may provide a pivotal indicator for common regulation and/or associated function of sets of genes.
We present the first gene cluster model capable of handling spatial data. Our model extends a popular computational model for gene cluster prediction, called *δ-teams*, from sequences to general graphs. In doing so, δ-teams are single-linkage clusters of a set of shared vertices between two or more undirected weighted graphs such that the largest link in the cluster does not exceed a given threshold δ in any input graph.
We apply our model to human and mouse data to find *spatial gene clusters*, i.e., gene sets with functional associations that exhibit close neighborhood in the spatial conformation of the chromosome across species.

Keywords: Spatial gene cluster · Gene teams · Single-linkage clustering · Graph teams · Hi-C data

1 Introduction

Distance-based clustering algorithms are paramount to approach various questions across all data-driven fields including comparative genomics. Here, we study the problem of discovering single-linkage clusters of a set of corresponding vertices (where correspondence is either provided through a bijective mapping or equivalence classes) between two or more undirected weighted graphs $G_1, \dots, G_k$ such that the largest link in the cluster (measured in terms of the weighted shortest path) does not exceed a given threshold δ in either graph G_i, $1 \leq i \leq k$. We call such clusters (δ-) teams, thereby adopting notation used by an extensive trail of literature that studies the equivalent problem on permutations and sequences [2,9,19,21].

A prominent use case of δ-teams in comparative genomics is the detection of *gene clusters*, which are sets of genes with associated functionality such as the encoding of different enzymes used in the same metabolic pathway. In many organisms instances exist where such genes are also locally close to each other in the genome, i.e., their positions fall within a narrow region on the same chromosome. They may even remain in close proximity over a longer evolutionary period, despite the fact that genomes regularly undergo mutations such as genome rearrangements, gene- or segmental duplications, as well as gene insertions and deletions. Such mutations may also affect the order and copy number of genes within a gene cluster. Molecular biologists argue that a conserved neighborhood is beneficial for co-regulation, as is true in the prominent case of *operons* in prokaryotes [10]. Gene clusters are also prevalent in eukaryotes, even in animals, where the HOX gene cluster is without doubt the best studied representative. HOX genes are transcription factors that regulate the embryological development of the metazoan body plan [12].

Yet, the function of many genes is often barely understood or entirely unknown despite the increasing number of whole genome data that is becoming available. Hence, a popular approach in comparative genomics is to work this way backwards, starting with the investigation of conserved gene proximity in genomes of a reasonably phylogenetically diverse set of species. Here, the underlying assumption is made that accumulated genome rearrangements will have shuffled the genome sequences sufficiently so that natural selection becomes a plausible cause of conserved gene neighborhoods. By identifying homologous sets of genes that are consistently close to each other across several species, candidate gene clusters are identified that are then subject to more thorough functional analysis.

More recently, new technologies emerged, allowing the study of the spatial structure of genomes. *High-throughput chromosome conformation capture* (Hi-C), the most popular among such approaches, allows assessing the conformation of the chromatin structure in a cell sample through measuring the number of observed contacts between DNA regions [3]. The Hi-C method makes use of formaldehyde to covalently bond proteins and DNA strings which are located next to each other in the cell. After crosslinking, the cells are lysed and the DNA is digested by a restriction enzyme. Digested fragments bonded by the same protein are ligated. Sequencing the hybrid sequences reveals three-dimensional contacts between their genomic origins. The outcome of the experiment is a table, called *Hi-C map*, that records observed contacts. Each row and each column of the Hi-C map represents an equally sized segment of the genome sequence, and a count in each cell indicates how often sequences of the corresponding segments have been observed during the experiment. The size of these segments is known as *resolution*. It is a crucial parameter regarding the quality of the data. The higher the resolution of the chromatin structure is, the smaller is the segment size, but also the more data is needed to get significant results. An increasing number of Hi-C maps has recently been made publicly available (human and mouse [8], fruit fly [16]) and is used to answer numerous biological questions, starting from gene regulation and replication timing [8,13] to genome scaffolding and haplotyping [4,15].

Gene cluster discovery has sparked the development of various computational models for identifying sets of genes that exhibit close proximity. Such models typically rely on abstract data structures known as *gene order* sequences, which describe the succession of genes in chromosomes. In doing so, each element of a gene order sequence is the identifier of a gene's associated gene family. A popular method to find gene clusters is based on the identification of *common intervals* in these sequences, which are intervals with an identical set of elements (i.e. gene family identifiers), independent of the elements' order and multiplicity [7,14,18]. Since their first mentioning in [18], common intervals became the source for several generalizations [11,22], among others, *δ-teams* [9]. δ-Teams are sets of elements where the distance between any two successors across all sequences is bounded by a given threshold $\delta \geq 0$. This flexible model not only facilitates the detection of gene clusters that are interspersed by unrelated inserted genes, but also allows the consideration of distance in terms of nucleotide base pairs between genes.

Even more than gene proximity on linear DNA sequences, the spatial conformation of chromosomes may provide a pivotal indicator for common regulation and/or associated function of sets of genes. Evidence of *spatial gene clusters* has been put forward already by Thévenin *et al.* [17] who studied spatial proximity within functional groups of genes in the human genome. In this work, we present the first spatial gene cluster model. Our model extends the δ-teams model from sequences to undirected weighted graphs, facilitating the detection of genes that are consistently spatially close in multiple species. In doing so, our method integrates Hi-C and genome sequence data into weighted undirected graphs, where vertices represent gene family identifiers of genes and weighted edges correspond to distances obtained from Hi-C data.

The remainder of this manuscript is organized as follows: In Sect. 2 we formally define δ-teams on graphs and present an algorithm for their discovery. We then extend our approach to finding δ-teams *with families*, i.e., the case where vertices across graphs are related through a common family membership, allowing multiple members of the same family to be part of the same graph. In Sect. 3 we show how δ-teams can be used to find candidate sets of spatial gene clusters using a combination of genome and Hi-C data of two or more species. We evaluate our approach using data from human and mouse. Section 4 concludes this manuscript and provides an outlook on future work.

An implementation of our method in the `Python` programming language is available from `http://github.com/danydoerr/GraphTeams`.

2 Discovering δ-Teams in Graphs with Shared Vertex Sets

In this section we discuss the general problem of identifying single linkage clusters in a collection of graphs, where the largest link does not exceed a given distance threshold δ. We call such clusters δ-teams to remain in line with previous literature which studied the equivalent problem on permutations and sequences.

To simplify presentation, we describe only the case of two input graphs G and H in detail. The general case can be trivially inferred. In fact, our implementation (see Sect. 3) supports comparison of two or more graphs.

We study undirected graphs $G = (V, E)$ with distance measure $d_G : V \times V \rightarrow \mathbb{R}_{>0}$. While subsequent definitions adhere to the general case, for all our purposes we assume edge-weighted graphs and use as distance measure the length of the shortest path between any two vertices, measured by the sum of the path's edge weights if such exists and ∞ otherwise. We use $E(G), V(G)$ to denote the edge and vertex set of a graph G, respectively. Since we will refer frequently to sets of vertices in one of several graphs, we will indicate the origin of a vertex set $S \subseteq V(G)$ of a graph G through subscript notation, i.e. S_G, whenever this information is relevant. We are interested in sets of vertices that are connected through paths on which the distance between two successive members is bounded by δ:

Definition 1 ($\boldsymbol{\delta}$-set). *Given a graph G with distance measure d_G and a threshold value $\delta \geq 0$, a vertex set $S \subseteq V(G)$ is a* δ-set *if for each pair of vertices $u, v \in S$ there exists a sequence $P = (u, \ldots, v) \subseteq S$ such that the distance $d_G(w, z)$ between any two consecutive vertices w and z of P is less than or equal to δ.*

The subsequent definitions establish relations between δ-sets across two graphs G and H with shared vertex set $V_\cap = V(G) \cap V(H)$. In doing so, we assume that there is a common non-empty set of vertices between the two graphs that is subject to subsequent analysis. Vertices that are unique to either of the two graphs are disregarded, yet may be relevant due to their involvement in paths between common vertices.

Definition 2 ($\boldsymbol{\delta}$-cluster). *Given two graphs G and H with distance measure d_G and d_H, respectively, and a threshold value $\delta \geq 0$, a vertex set $S \subseteq V_\cap$ is a* δ-cluster *if it is a δ-set in both G and H under distance measures d_G and d_H, respectively.*

Definition 3 ($\boldsymbol{\delta}$-team). *Given two graphs G and H, a δ-cluster S of G and H is a* δ-team *if it is* maximal*, i.e., there is no δ-cluster S' of G and H such that $S \subsetneqq S'$.*

Example 1. The two graphs G and H depicted in Fig. 1 (a) have three δ-teams: 1-team $\{d, f\}$; 2-team $\{c, d, f\}$, and 3-team $\{a, c, d, f, g\}$. The set $\{c, d, f, g\}$ exemplifies a non-maximal 3-cluster of G and H.

2.1 Finding $\boldsymbol{\delta}$-Teams by Decomposing Graphs with Divide-and-Conquer

Given the above definitions, the following computational problem naturally arises and is subject to this work:

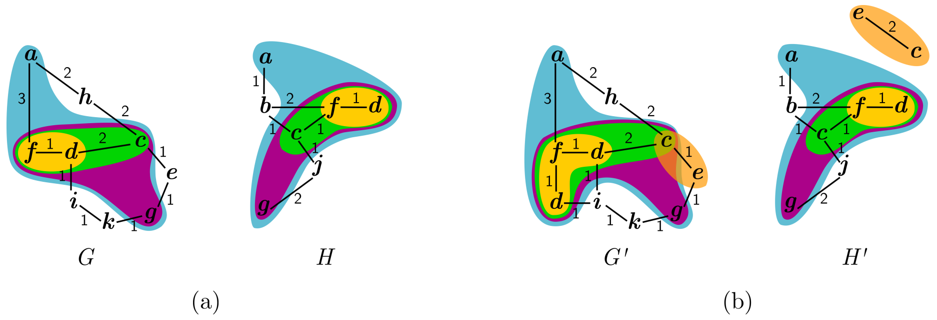

Fig. 1. Examples of δ-teams and δ-clusters in graphs without families (a) and with families (b). δ-teams and -clusters are highlighted by areas of shared color. Edge labels indicate weights. Vertices in Figure (b) are represented by their family identifier.

Problem 1. Given two graphs G and H with distance measure d_G and d_H, respectively, and a threshold value $\delta \geq 0$, find all δ-teams of G and H.

The first observation that is key to addressing the problem at hand, is that two δ-teams cannot overlap. The following lemma, in which $\text{Teams}_\delta(S)$ denotes the set of δ-teams of vertex set S, is basis to all permutation-based (gene-) team algorithms and holds true for the here proposed generalization, too (the disjoint union of two sets is denoted by $\uplus$):

Lemma 1. *[2,9] Given two graphs G and H with common vertex set $V_\cap$ and a threshold value $\delta \geq 0$, there exists a partition $\{V', V''\}$ of $V_\cap = V' \uplus V''$ such that* $\text{Teams}_\delta(V_\cap) = \text{Teams}_\delta(V') \uplus \text{Teams}_\delta(V'')$.

The lemma leads to a simple divide-and-conquer approach which has already been applied by He and Goldwasser [9] for the restricted case of sequential data. Here, we apply this lemma to general graphs. Algorithm Decompose divides the common vertex subset $S \subseteq V_\cap$ of graphs G and H into smaller subsets as long as S is not a δ-set in both graphs.

Algorithm 1 Decompose(S)

```
Input: graphs $G$, $H$; vertex subset $S \subseteq V_\cap$, $S \neq \emptyset$; threshold value $\delta \geq 0$
Output: all $\delta$-teams of $G$ and $H$ within $S$
1: $S' \leftarrow$ SmallMax($S, S$) // find a smaller maximal $\delta$-set $S' \subseteq S$ of $G$ or $H$
2: if $|S| = |S'|$ then
3:     return $\{S\}$
4: else
5:     return Decompose($S'$) $\cup$ Decompose($S \setminus S'$)
6: end if
```

Because Algorithm 1 proceeds from larger to smaller sets, a vertex set S, identified by the algorithm, that is a δ-set in both G and H is always maximal and therefore a δ-team. Procedure SMALLMAX (see line 1) finds a maximal δ-set S' smaller than S, or, if the smallest maximal δ-set (that is still a subset of S) in both G and H is S itself, returns S. This will be further elaborated in following section.

2.2 Identifying Maximal δ-Sets

Maximal δ-sets are identified by function SMALLMAX as described in pseudocode by Algorithm 2.

Algorithm 2 SMALLMAX(S_G, S_H)

```
Input: graphs $G$, $H$; vertex subsets $S_G \subseteq V(G)$ and $S_H \subseteq V(H)$; threshold value $\delta \geq 0$
Output: a maximal $\delta$-set $S'_G \subseteq S_G$ or $S'_H \subseteq S_H$
 1: choose random vertices $u \in S_G$ and $v \in S_H$
 2: initialize sets $S'_G = \{u\}$, $S'_H = \{v\}$
 3: initialize Boolean variables $p_G$, $p_H$ with True
 4: while ($p_G$ or $S'_G = S_G$) and ($p_H$ or $S'_H = S_H$) and ($p_G$ or $p_H$) do
 5:   for each graph $X = G, H$ do
 6:     if $\exists\ s \in S_X \setminus S'_X$ s.t. $\exists\ s' \in S'_X$ with $d_X(s, s') \leq \delta$ then
 7:       add vertex $s$ to set $S'_X$
 8:     else
 9:       $p_X \leftarrow$ False
10:     end if
11:   end for
12: end while
13: if ($\neg p_G$ and $S'_G \neq S_G$) then return $S'_G$ else return $S'_H$
```

Note that procedure SMALLMAX is drafted for a general setting that permits the discovery of different vertex sets in graphs G and H, respectively. In doing so, SMALLMAX can also be used in the case of finding δ-teams *with families* that is subject of Sect. 2.4. For now, the input sets S_G and S_H are identical.

SMALLMAX identifies a smaller maximal δ-set in either vertex set S_G or S_H. In each iteration (lines 4-12), the algorithm searches in each graph $X = G, H$ a vertex s of set S_X which has not been previously visited and that has distance at most δ from any already visited node. To this end, a list S'_X is maintained that keeps track of already visited vertices of set S_X. Boolean variables p_X indicate whether unvisited, yet reachable vertices in set $S_X \setminus S'_X$ could be found in graph X. The iteration is controlled by three different cases (line 4): If no unvisited node can be found, SMALLMAX has identified either a smaller δ-set of S_X or, if the traversal is exhausted, S_X itself. In the former case, the procedure stops and returns the visited subset S'_X of S_X. In the latter case, the algorithm continues the search for a smaller δ-set in the corresponding other vertex set S_Y, $Y = \{G, H\} \setminus X$, and will return such if found. Otherwise, the smallest maximal

δ-set in both S_G and S_H are the sets themselves. This also leads to a disruption of the while-loop (lines 4-12) and, by convention, the return of set S'_H $(= S_H)$.

Because SmallMax does not go further than distance δ from any already visited node of S'_X, it is clear that the returned vertex set is a δ-set. It is also maximal, because the algorithm does not stop prior to having found all vertices of S_X that can be reached from the starting node (which is also a member of S_X and S'_X).

The time complexity of algorithm Decompose depends on the number of its own recursive function calls. The decomposition of set S into sets S' and $S \setminus S'$ that is performed in line 5 of Decompose takes $O(|S|)$ time, but is overshadowed by the time complexity of SmallMax. For SmallMax, the most costly operation is the search for the next node s of $S_X \setminus S'_X$. This can be found through successive traversal of each graph using *breadth-first search* (BFS) outgoing from any arbitrary vertex of sets S_G and S_H, respectively. The BFS determines the running time of SmallMax and requires $O(|V(G)| + |E(G)| + |V(H)| + |E(H)|)$ time. In the worst case, Decompose needs $|V_\cap|$ iterations to decompose the initial, shared vertex set $V_\cap$.

This leads to an overall running time of $O(|V_\cap| \cdot (|V(G)| + |E(G)| + |V(H)| + |E(H)|))$ for Algorithm 1.

2.3 The Special Case of Shortest-Path Graphs

In the special case where each pair of vertices u, v of vertex set $V_\cap$ has a directly connecting edge whenever their distance is smaller or equal to δ, SmallMax takes $O(|V_\cap|)$ time in each iteration. This observation leads to an alternative approach for the general case that may in practice be faster for certain instances or applications: From the input graphs G, H two new graphs G' and H' are derived by computing shortest paths between all pairs of vertices in $V(G)$ and $V(H)$, respectively. In the new graph G' two vertices $u, v \in V(G) = V(G')$ are connected with an edge of weight 1 if their distance is smaller δ and, similarly, for graph H'. Then, the enumeration of δ-teams of G and H is equivalent to computing 1-teams in G' and H'. Our implementation includes an option for the computation of δ-teams using this alternative approach. Shortest paths are obtained with Floyd-Warshall's algorithm which has a running time of $O(|V|^3)$ [5].

2.4 δ-Teams with Families

Family labels allow correspondences between vertices of the input graphs G and H that go beyond 1-to-1 assignments, which is the scenario best suitable for our application as further explained in Sect. 3. Given a graph $G = (V, E)$, let $\mathcal{F} : V \to F$ be a surjective mapping between vertices and families.

We extend the concepts of δ-cluster and δ-team to families as follows:

Definition 4 (δ-cluster with families). *Given two graphs G and H with distance measures d_G and d_H, respectively, a family mapping $\mathcal{F}$ and a threshold value $\delta \geq 0$, a pair of vertex sets (S_G, S_H) with $S_G \subseteq V(G)$ and $S_H \subseteq V(H)$,* *is a* δ-cluster *if (i)* $\mathcal{F}(S_G) = \mathcal{F}(S_H)$ *and (ii)* S_G *and* S_H *are* δ*-sets in* G *and* H *under distance measures* d_G *and* d_H*, respectively.*

Definition 5 (δ-team with families). *Given two graphs* G *and* H*, a* δ*-cluster* (S_G, S_H) *of* G *and* H *is a* δ-team *if it is* maximal*, i.e., there is no* other δ*-cluster* (S'_G, S'_H) *of* G *and* H *such that* $S_G \subseteq S'_G$ *and* $S_H \subseteq S'_H$*.*

Example 2. The two graphs G' and H' depicted in Fig. 1 (b) have four δ-teams that are in the following represented by their family set: 1-team $\{d, f\}$; 2-teams $\{c, d, f\}$ and $\{c, e\}$, and 3-team $\{a, c, d, f, g\}$. The set $\{c, d, f, g\}$ exemplifies a non-maximal 3-cluster of G' and H'.

With the generalization to families, Lemma 1 is no longer applicable. However, Wang *et al.* [19] provide an adaptation which shows that the original divide-and-conquer approach can be trivially extended:

Lemma 2. *[19] Given two graphs* G *and* H*, a family mapping* $\mathcal{F}$ *and a threshold value* $\delta \geq 0$*, let* $S_G \subseteq V(G)$*,* $S_H \subseteq V(H)$*, s.t.* $\mathcal{F}(S_G) = \mathcal{F}(S_H)$ *and* B *be a maximal* δ*-set of* S_G *or* S_H*. W.l.o.g. let* $B \subseteq S_G$*, then* $\text{Teams}_\delta(S_G, S_H) = \text{Teams}_\delta(B, S'_H) \cup \text{Teams}_\delta(S_G \setminus B, S''_H)$*, where* $S'_H = \{v \in S_H \mid \mathcal{F}(v) \in \mathcal{F}(B)\}$ *and* $S''_H = \{v \in S_H \mid \mathcal{F}(v) \in \mathcal{F}(S_G \setminus B)\}$*.*

The adaptations to algorithm Decompose are a straightforward implementation of Lemma 5 and are shown in Algorithm 3 (DecomposeFamilies).

Algorithm 3 DecomposeFamilies(S_G, S_H)

```
Input: Graphs $G$ and $H$, mapping $\mathcal{F}$, vertex sets $S_G \subseteq V(G)$ and $S_H \subseteq V(H)$ such
    that $\mathcal{F}(S_G) = \mathcal{F}(S_H) \neq \emptyset$, distance measures $d_G, d_H$, and $\delta \geq 0$
Output: all $\delta$-teams of $G$ and $H$ that are subset or equal to $(S_G, S_H)$
    // find a maximal $\delta$-set $S'_X \subseteq S_X$, where $X$ is a placeholder for graph $G$ or $H$
1: $S'_X \leftarrow$ SmallMax$(S_G, S_H)$
2: if $S_X = S'_X$ then
3:     return $\{(S_G, S_H)\}$
4: else
5:     $Y \leftarrow \{G, H\} \setminus X$
6:     $S'_Y \leftarrow \{v \in S_Y \mid \mathcal{F}(v) \in \mathcal{F}(S'_X)\}$
7:     $S''_Y \leftarrow \{v \in S_Y \mid \mathcal{F}(v) \in \mathcal{F}(S_X \setminus S'_X)\}$
8:     return DecomposeFamilies$(S'_X, S'_Y) \cup$ DecomposeFamilies$(S_X \setminus S'_X, S''_Y)$
9: end if
```

To efficiently retrieve vertices associated with families of $\mathcal{F}(S'_X)$ and $\mathcal{F}(S_X \setminus S'_X)$ (see lines 6 and 7 of Algorithm 3), we follow Wang *et al.* [19] and maintain a table of linked lists that maps family identifiers with its members in each respective graph. $\mathcal{F}(S'_X)$ can be built in $O(|S'_X|)$ time while $\mathcal{F}(S_X \setminus S'_X)$ needs $O(|S_X|)$ time. Afterwards, it is possible to build S'_Y and S''_Y in $O(|S_Y|)$ time. The runtime of SmallMax remains the same for Algorithm 3. Yet, because the input sets S_G and S_H can no longer be decomposed into disjoint sets, Algorithm 3 requires overall $O((|V(G)| + |E(G)|) \cdot (|V(H)| + |E(H)|))$ time and $O(|V(G)| + |E(G)| + |V(H)| + |E(H)|)$ space.

3 Application to Spatial Gene Cluster Discovery

We will now show how the discovery of δ-teams with families allows to find spatial gene clusters in genomic data of two or more species. We implemented Algorithm 3 in the `Python` programming language and provide an entirely automated `Snakemake` workflow for the identification of spatial gene clusters. Our workflow takes as input the fully assembled sequences of a collection of genomes as well as their corresponding Hi-C maps. It normalizes the Hi-C maps, establishes relationships between Hi-C segments and genes, and constructs weighted graphs that are then input to Algorithm 3. Further, our workflow allows the computation of a ranking scheme for gene cluster candidates based on shared functional associations of their members when provided with additional GO-annotations. Our approach is, to the best of our knowledge, the first of its kind that is capable of identifying spatial gene clusters. Our `Snakemake` workflow can be obtained from `http://github.com/danydoerr/GraphTeams`.

3.1 Constructing Graphs from Genome Sequences and Hi-C Data

For each genome, we construct an undirected weighted graph in which vertices correspond to genes that are labeled with the identifier of their associated gene family and in which weighted edges correspond to distances obtained from the contact counts of the genomes' respective Hi-C maps. Then, δ-teams (with families according to the genes' families) in the constructed graphs will correspond to spatial gene cluster candidates.

We first map the Hi-C data onto their chromosomal sequences. In doing so, we associate genes with segments of the Hi-C map. Consequently, contact counts between genes correspond to the contact counts of their associated segments. The value of a contact count does not represent a distance but a closeness score, hence a transformation is needed. We define the *distance* between two genes g_i, g_j associated with Hi-C map M as

$$d_M(g_i, g_j) = \max_{k,l}(M_{kl}) + 1 - M_{ij} \; . \tag{1}$$

Hi-C maps are symmetric matrices, therefore $d_M(g_i, g_j) = d_M(g_j, g_i)$. Whenever two gene pairs fall into the same Hi-C segment, their distance is estimated by incorporating their proximity on the DNA sequence. To this end, each base pair between the midpoints of two genes is scored with a relative contact count of C/r, where C is the average contact count between two adjacent segments in the Hi-C map, i.e., the mean of $M_{i,i+1}$ of Hi-C map M, and r is the resolution of the Hi-C map, i.e., the size of its segments. This estimator works well for our purposes because Hi-C data shows strong correlations with distances on the DNA sequence.

It is common that Hi-C maps contain large numbers of empty cells as a result of erroneous measurements and deliberate blanking of the contact counts around the centromere. We do not apply any correction to such cells except to those that correspond to adjacent segments, i.e., the $M_{i,i+1}$ cells. Here, we use the same estimator as described above for genes falling into the same cell of the Hi-C map.

Because we will compare distances obtained from different Hi-C maps, we must ensure that they all use the same scale. We do this by multiplying all distances of each Hi-C map M with a factor $c/(\max_{k,l}(M_{kl})+1)$ where c is the *average maximum contact count* across all Hi-C maps.

3.2 Quantifying Functional Associations of Gene Clusters using Gene Ontology Annotations

We quantify functional associations between genes of a gene cluster candidate by testing against the *null* hypothesis that a gene in a gene cluster is functionally not more associated to one of its co-members than to any other genes in the genome. To this end, we make use of *Gene Ontology* (GO) [1] annotations and relate between gene functions by means of the gene ontology hierarchy that corresponds to the domain "Biological Processes". In doing so, we measure *GO-based functional dissimilarity* (GFD) [6] between pairs of GO-annotated genes. Given a directed acyclic graph $G = (V, E)$ corresponding to a GO-hierarchy, $r_G(g) = \{v \in V(G) \mid g \text{ associated with } v\}$ denotes the set of *GO terms*, i.e., vertices of the GO hierarchy G, with which gene g is associated. Further, $p_G(u, v)$ denotes the length of the shortest path between two vertices $u, v \in V$ measured in the number of separating nodes. The GFD between two GO-annotated genes g and g' is then defined as

$$gfd_G(g, g') = \min_{(u,v)\in r_G(g)\times r_G(g')} \left(\frac{p_G(u, v)}{depth_G(u) + depth_G(v)} \right), \qquad (2)$$

where $depth_G(w)$ is the length of the path from the root vertex of G to vertex w. This measure gives then rise to the *gene cluster penalty* defined for a gene set $C \subseteq \mathcal{G}$ of a genome $\mathcal{G}$ as follows:

$$\phi_G(C, \mathcal{G}) = \sum_{g\in C} \left(\min_{g'\in C\setminus g} gfd(g, g') - \min_{g''\in \mathcal{G}\setminus g} gfd(g, g'') \right). \qquad (3)$$

In our analysis, we rank gene clusters according to p-values empirically computed from sample pools of size of 10^7 which are drawn for each gene cluster size, respectively.

3.3 Finding Candidates of Spatial Gene Clusters in Human and Mouse

We used the approach described in Section 3.1 to find spatial gene cluster candidates in human and mouse. To this end, we queried the Ensemble Genome Browser (release 88) [23] to obtain information about orthologous genes of the human reference sequence `GRCh38.p10` and the mouse reference sequence `GRCm38.p5`. The obtained data consists of 19,843 human genes that are orthologous to 20,647 mouse genes. The intra-chromosomal Hi-C maps of the human and mouse genomes that we use in this study were first published in Dixon *et al.* [8] and have a resolution of 40 kb.

The graphs for the human and mouse datasets were constructed as previously described. Subsequently, we used them to find δ-teams for different values of δ. All computations were performed on a Dell RX815 machine with 64 2.3 GHz AMD Opteron processors and 512 GB of shared memory. The running times for computing all δ-teams for each value of δ are shown in the bottom right plot of Fig. 2 and range from 62 minutes for $\delta = 200$ to 111 minutes for $\delta = 600$. The plot indicates a sharp increase of running time for $\delta > 400$ that correlates with the increase of the size of identified δ-teams in our dataset.

Apart from the performance of our algorithm, we also wanted to investigate how suitable spatial data is to improve the search for gene clusters. Next to the graphs that were generated as described in Sect. 3.1 and which we will further call *spatial graphs*, we constructed a second type of graphs, called *sequential graphs*. In the latter, distances between genes are limited to those that are adjacent in their respective genome sequences. Connecting these adjacent genes, we used the same distances as in the former graph so that distances between both types of graphs are comparable. Since our algorithm is a direct generalization of previous methods acting on linear DNA sequences, we can generate results of these methods using sequential graphs. We call δ-teams that are found in spatial graphs *3D gene clusters*, whereas those in sequential graphs are called *1D gene clusters*.

Figure 2 shows the results for both graphs. In the plot on the top left, we can see that the number of gene clusters grows for both types of graphs with increasing values of δ while the number of 3D gene clusters is slightly higher than that of 1D gene clusters. This changes after $\delta = 350$ when more 3D gene clusters are merged than new instances are found, leading to a rapid decrease in their number along with an increase in their size (see plot on the top right). The peak associated with this phenomenon is delayed in the sequential graphs, owing to the fact that genes are there more stretched out. This is also the reason why we find that some 1D gene clusters are much denser in the spatial graphs. More surprisingly, we also find gene clusters that can only be found in spatial graphs for a given threshold value δ. We call the average amount of genes in a cluster that can be found in the spatial graphs, but not in the sequential ones *spatial gain* (see plot at the bottom left). We see an increase in spatial gain around $\delta = 250$ until a saturation seems to be reached at $\delta = 450$.

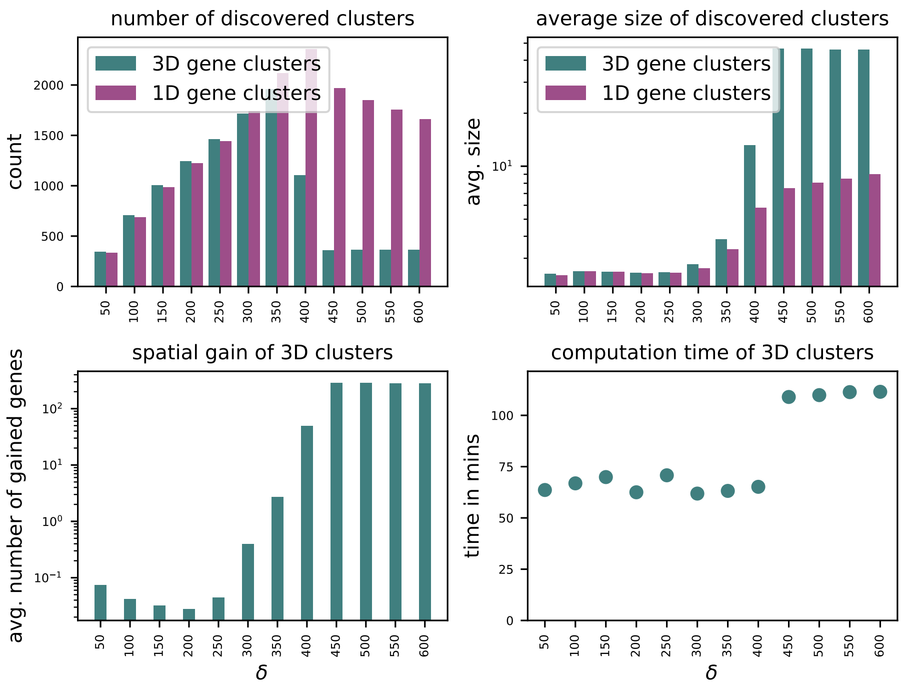

Fig. 2. Results of Algorithm 3 for different values of δ. Graphs are constructed from Hi-C datasets of human and mouse. The plots show for each threshold value δ, the number of discovered clusters (upper left) and their average sizes (upper right) in the spatial and sequential graphs, respectively, the average number of gained genes in the 3D gene clusters versus the 1D gene clusters (lower left), and the computation time for the 3D gene clusters (lower right).

We further investigated gene clusters discovered with $\delta = 350$, which strike a fair balance between number and size as can be readily observed from our previous analysis. The datasets of both, 3D and 1D gene clusters, were used to evaluate functional associations between gene cluster members. To this end, GO-annotations of the human genome were obtained from [1] to compute gene cluster penalties and to rank gene clusters according to their empirical p-value as described in Section 3.2. In the obtained gene ontology dataset, 15,737 out of 19,843 human genes were associated with one or more GO-terms. Because the analysis is restricted to those genes with annotated GO-terms, only 1,559 out of 1,961 3D gene clusters and 1,669 out of 2,118 1D gene clusters could be further investigated. 18.54% of the 3D gene clusters and 18.33% of the 1D gene clusters exhibited a significant empirical p-value for $p < 0.05$. Overall, significant 3D gene clusters tend to include more (annotated) genes (total: 930) than their 1D counterparts (total: 886). Table 1 lists the top 20 3D gene clusters that are either not found in the set of significant 1D gene clusters, or only partially found, or broken into two or more sub-clusters. We can see that many of them are already known from the literature. E.g., we find four clusters of *olfactory receptor* (OR) genes on different chromosomes, the *taste receptor type 2* (TAS2R) gene cluster and the HOXC gene cluster. The latter is one of three clusters among the top 20 but can be found in the 1D results only as a composition of sub-clusters. Therefore, these genes seem to be even closer together in 3D than on the DNA strand. The same is true for other clusters, such as that of the *testis-specific protein Y-encoded* (TSPY) and *superfamily Ig belonging lectins* (SIGLEC) which were not even partially detected in the 1D graphs.

Table 1. Top 20 3D gene clusters with smallest p-value. Clusters that can be found as split sub-clusters in the 1D results are marked by an asterisk. Those completely absent in the 1D results are marked by a plus.

Name	**Genes**	**Penalty**	**p-Value**
HOXC*	HOTAIR_2, HOTAIR_3, HOXC10, HOXC11, HOXC12, HOXC13, HOXC4, HOXC5, HOXC6, HOXC8, HOXC9	0.006	$1 \cdot 10^{-7}$
OR	OR5AP2, OR5AR1, OR5M1, OR5M10, OR5M11, OR5M3, OR5M8, OR5M9, OR5R1, OR8K1, OR8U1, OR9G1, OR9G	0	$1 \cdot 10^{-7}$
IGHV*	IGHV3-11, IGHV3-13, IGHV3-20, IGHV3-21, IGHV3-23, IGHV3-30, IGHV3-33, IGHV3-35, IGHV3-64D, IGHV3-7	0	$1 \cdot 10^{-7}$
KRTAP*	KRTAP13-1, KRTAP13-2, KRTAP13-3, KRTAP13-4, KRTAP15-1, KRTAP24-1, KRTAP26-1, KRTAP27-1	0	$1 \cdot 10^{-7}$
TAS2R	TAS2R14, TAS2R19, TAS2R20, TAS2R31, TAS2R46, TAS2R50	0	$3.70 \cdot 10^{-6}$
OR	OR2A12, OR2A14, OR2A25, OR2A5	0	$9.09 \cdot 10^{-5}$
ZSCAN4	NKAPL, ZKSCAN3, ZKSCAN4, ZSCAN26	0.006	0.00015
TRAV	TRAV12-1, TRAV12-2, TRAV12-3, TRAV13-1, TRAV13-2, TRAV17, TRAV18, TRAV19, TRAV22, TRAV23DV6, TRAV5, TRAV8-1, TRAV8-3, TRAV9-2	0	0.00037
OR	OR5AC1, OR5H1, OR5H14	0	0.00037
IGHV$^+$	IGHV1-18, IGHV1-24, IGHV1-3	0	0.00037
BTN3$^+$	BTN3A1, BTN3A2, BTN3A3	0	0.00037
(unnamed)	GTF2A1L, STON1, STON1-GTF2A1L	0	0.00037
CYP3A	CYP3A4, CYP3A43, CYP3A5, CYP3A7, CYP3A7-CYP3A51P	0.028	0.00037
(unnamed)	ADGRE1, C3, CD70, GPR108, TNFSF14, TRIP10, VAV1	0.057	0.00047
ZNF	CCDC106, FIZ1, U2AF2, ZNF524, ZNF580, ZNF784, ZNF865	0.097	0.00110
OR	OR8B12, OR8B4, OR8B8	0.012	0.00376
KIR	KIR2DL1, KIR2DL3, KIR2DL4, KIR2DS4, KIR3DL1, KIR3DL2, KIR3DL3	0.179	0.00243
MMP	MMP12, MMP13, MMP3	0.035	0.00486
TSPY$^+$	TSPYL1, TSPYL4	0	0.00504
SIGLEC$^+$	SIGLEC12, SIGLEC8	0	0.00504

4 Discussion and Outlook

The enumeration of common intervals in sequences has been subject to various extensions including δ-teams. Here, we described a generalization of δ-teams from sequences to graphs. We presented a novel algorithm for the enumeration of δ-teams that, when trivially extended to k graphs $G_i = (V_i, E_i)$, for $i = 1, \dots, k$, will run in $O(\sum_i |V_i| + |V_\cap| \cdot \sum_i(|V_i| + |E_i|))$ time and $O(\sum_i(|V_i| + |E_i|))$ space, where $V_\cap = V_1 \cap \cdots \cap V_k$. Our algorithm beats the naive approach that requires $O(\sum_i |V_i|^3)$ time and $O(\sum_i |V_i|^2)$ space by computing all-pairs-shortest-paths and then using a standard single linkage clustering algorithm to enumerate δ-teams. Further, we provide an algorithm for the computation of δ-teams that, when trivially extended to k graphs *with families*, will run in $O(k \cdot \prod_i(|V_i| + |E_i|))$ time and $O(k \cdot \sum_i(|V_i| + |E_i|))$ space.

In comparison, the best algorithm for the enumeration of δ-teams in k permutations of size n runs in $O(k \cdot n \cdot \log N)$ time, where N denotes the number of reported δ-teams [20]. The best algorithm that solves the corresponding family-based problem for k sequences of lengths $n_1, \dots, n_k$ runs in $O(k \cdot C \cdot \log(n_1 \cdots n_k))$ time, where C is a factor accounting for the number of possible 1:1 assignments between family members across the k graphs [19]. The differences in running time between the permutation-, sequence- and our graph-based algorithms reflect the fact that the latter solve much harder problems. Nevertheless, further studies may lead to improved algorithms. It seems possible that the problem of finding δ-teams in graphs *without families* could be solved faster with the help of a guide tree that allows to find a maximal δ-set by traversing each graph in fewer steps than required by an exhaustive graph traversal. Alternatively, a randomized variant of our algorithm could assert a better expected running time.

The presented algorithmic work could also be extended into another direction, by allowing the direct computation of the single-linkage hierarchy. This makes the gene cluster analysis no longer dependent on a fixed δ, but will provide all possible δ-clusters through a single computation. This idea has also been applied for δ-teams in sequences, where the hierarchy is called *gene team tree* [21,24].

By identifying δ-teams *with families*, we provide a flexible model that is well suitable to capture the complexity of biological datasets such as those at hand. Our presented algorithm and our implementation are fast enough to conveniently process large graphs as demonstrated in the evaluation of this study. The identification of all δ-clusters in the studied Hi-C dataset of human and mouse took between 62 and 111 minutes on state-of-the-art hardware.

Finally, we evaluated functional associations between members of 3D and 1D gene cluster candidates, respectively. Our experimental evaluation provides further evidence for the existence of spatial gene clusters, that is, sets of functionally associated genes whose members are closer to each other in the 3D space than on the chromosomal sequence.

Acknowledgements

We are very grateful to Krister Swenson for kindly providing the Hi-C data used in this study and for his many valuable suggestions. We wish to thank Pedro Feijão for many fruitful discussions in the beginning of this project. This work was partially supported by DFG GRK 1906/1.

References

1. Ashburner, M., Ball, C.A., Blake, J.A., Botstein, D., Butler, H., Cherry, J.M., Davis, A.P., Dolinski, K., Dwight, S.S., Eppig, J.T., Harris, M.A., Hill, D.P., Issel-Tarver, L., Kasarskis, A., Lewis, S., Matese, J.C., Richardson, J.E., Ringwald, M., Rubin, G.M., Sherlock, G.: Gene ontology: tool for the unification of biology. The Gene Ontology Consortium. Nat. Genet. 25(1), 25–29 (2000)
2. Beal, M., Bergeron, A., Corteel, S., Raffinot, M.: An algorithmic view of gene teams. Theor Comput Sci 320(2-3), 395–418 (2004)
3. Belton, J.M., McCord, R.P., Gibcus, J.H., Naumova, N., Zhan, Y., Dekker, J.: Hi–c: a comprehensive technique to capture the conformation of genomes. Methods 58(3), 268–276 (2012)
4. Burton, J.N., Adey, A., Patwardhan, R.P., Qiu, R., Kitzman, J.O., Shendure, J.: Chromosome-scale scaffolding of de novo genome assemblies based on chromatin interactions. Nat biotechnol 31(12), 1119–1125 (2013)
5. Cormen, T.H., Leiserson, C.E., Rivest, R.L.: Introduction to Algorithms. MIT Press, Cambridge, Massachusetts (1990)
6. Díaz-Díaz, N., Aguilar-Ruiz, J.S.: Go-based functional dissimilarity of gene sets. BMC Bioinformatics 12(1), 360 (2011)
7. Didier, G., Schmidt, T., Stoye, J., Tsur, D.: Character sets of strings. J Discrete Algorithms (Amst) 5(2), 330–340 (Dec 2006)
8. Dixon, J.R., Selvaraj, S., Yue, F., Kim, A., Li, Y., Shen, Y., Hu, M., Liu, J.S., Ren, B.: Topological domains in mammalian genomes identified by analysis of chromatin interactions. Nature 485(7398), 376–380 (2012)
9. He, X., Goldwasser, M.H.: Identifying conserved gene clusters in the presence of homology families. J. Comput. Biol. 12(6), 638–656 (Jun 2005)
10. Jacob, F., Perrin, D., Sanchez, C., Monod, J.: Operon: a group of genes with the expression coordinated by an operator. C. R. Hebd. Seances Acad. Sci. 250, 1727–1729 (Feb 1960)
11. Jahn, K.: Efficient computation of approximate gene clusters based on reference occurrences. J. Comput. Biol. 18(9), 1255–1274 (Aug 2011)
12. Larroux, C., Fahey, B., Degnan, S.M., Adamski, M., Rokhsar, D.S., Degnan, B.M.: The NK Homeobox Gene Cluster Predates the Origin of Hox Genes. Curr Biol 17(8), 706–710 (Apr 2007)
13. Ryba, T., Hiratani, I., Lu, J., Itoh, M., Kulik, M., Zhang, J., Schulz, T.C., Robins, A.J., Dalton, S., Gilbert, D.M.: Evolutionarily conserved replication timing profiles predict long-range chromatin interactions and distinguish closely related cell types. Genome res 20(6), 761–770 (2010)
14. Schmidt, T., Stoye, J.: Gecko and GhostFam: rigorous and efficient gene cluster detection in prokaryotic genomes. Methods Mol. Biol. 396(Chapter 12), 165–182 (2007)
15. Selvaraj, S., Dixon, J.R., Bansal, V., Ren, B.: Whole-genome haplotype reconstruction using proximity-ligation and shotgun sequencing. Nat biotechnol 31(12), 1111–1118 (2013)
16. Sexton, T., Yaffe, E., Kenigsberg, E., Bantignies, F., Leblanc, B., Hoichman, M., Parrinello, H., Tanay, A., Cavalli, G.: Three-dimensional folding and functional organization principles of the drosophila genome. Cell 148(3), 458 – 472 (2012)
17. Thevenin, A., Ein-Dor, L., Ozery-Flato, M., Shamir, R.: Functional gene groups are concentrated within chromosomes, among chromosomes and in the nuclear space of the human genome. Nucleic Acids Res. 42(15), 9854–9861 (Sep 2014)

18. Uno, T., Yagiura, M.: Fast Algorithms to Enumerate All Common Intervals of Two Permutations. Algorithmica 26(2), 290–309 (Feb 2000)
19. Wang, B.F., Kuo, C.C., Liu, S.J., Lin, C.H.: A New Efficient Algorithm for the Gene-Team Problem on General Sequences. TCBB 9(2), 330–344 (2012)
20. Wang, B.F., Lin, C.H.: Improved algorithms for finding gene teams and constructing gene team trees. TCBB 8(5), 1258–1272 (2010)
21. Wang, B.F., Lin, C.H., Yang, I.T.: Constructing a Gene Team Tree in Almost O(n lg n) Time. TCBB 11(1), 142–153 (2014)
22. Winter, S., Jahn, K., Wehner, S., Kuchenbecker, L., Marz, M., Stoye, J., Böcker, S.: Finding approximate gene clusters with Gecko 3. Nucleic Acids Res. 44(20), 9600–9610 (Nov 2016)
23. Yates, A., Akanni, W., Amode, M.R., Barrell, D., Billis, K., Carvalho-Silva, D., Cummins, C., Clapham, P., Fitzgerald, S., Gil, L., Girn, C.G., Gordon, L., Hourlier, T., Hunt, S.E., Janacek, S.H., Johnson, N., Juettemann, T., Keenan, S., Lavidas, I., Martin, F.J., Maurel, T., McLaren, W., Murphy, D.N., Nag, R., Nuhn, M., Parker, A., Patricio, M., Pignatelli, M., Rahtz, M., Riat, H.S., Sheppard, D., Taylor, K., Thormann, A., Vullo, A., Wilder, S.P., Zadissa, A., Birney, E., Harrow, J., Muffato, M., Perry, E., Ruffier, M., Spudich, G., Trevanion, S.J., Cunningham, F., Aken, B.L., Zerbino, D.R., Flicek, P.: Ensembl 2016. Nucleic Acids Res. 44(D1), D710 (2016)
24. Zhang, M., Leong, H.W.: Gene Team Tree - A Hierarchical Representation of Gene Teams for All Gap Lengths. J. Comput. Biol. 16(10), 1383–1398 (2009)